Apple IIGS
#52: Loading and Special Memory

Revised by: Dave Lyons May 1992
Written by: Eric Soldan January 1989

This Technical Note discusses strategies for preventing applications from loading into special memory.
Changes since July 1989: Noted that the System 6 Loader always tries non-special memory before special memory.

The System 6.0 Loader always tries to load segments into non-special memory before allowing them to load into special memory. The rest of this Note is useful if your application does not require System 6, or if you need an example of an initialization segment.

The System Loader loads your application starting at the lowest memory location possible. If you allow your program to load into special memory, the Loader first tries bank $01. If your program cannot load into special memory, it starts at bank $02. Either way, the Loader progresses to higher banks, and eventually, it may even try loading into bank $E1, which contains the super hi-res screen.

The problem with allowing your application to load into special memory is that the super hi-res screen is part of special memory. If you have a desktop application, part of your application may load into the super hi-res screen, and when you try to start QuickDraw II, it fails because the screen memory is already allocated.

When QuickDraw II fails because your program loaded into the SHR screen, it seems reasonable to assume that the Loader put your program there because it needed the RAM which special memory provides. This logic seems to make sense, but it is not completely reliable. The Loader (in System Software earlier than 6.0) tries to put your program into special memory **before** it tries purging dormant applications. This means that the more programs that run from the Finder that set the GS/OS or ProDOS 16 "restartable from memory" bit, the more likely it is that the next application launched that can load into special memory will load into the super hi-res screen.

For this reason, it is important not to let your application load into special memory, or at least not load into the super hi-res screen. If your application is not allowed to load into special memory, then the Loader will purge other dormant applications to make space for yours. One way to accomplish this is when linking your application. You can set the "no special memory" bit in the OMF `KIND` field of applications using OMF 2.0 or later, but this also prohibits your application from using bank $01.

Another way to avoid loading into the super hi-res screen is to have your initial segment allocate the super hi-res screen. You can accomplish this by starting QuickDraw II in your initial segment, then the rest of your program cannot load into the already-allocated super hi-res screen. This strategy could fail if the initial segment loaded into the super hi-res screen, but this is very unlikely and can be prevented by flagging the initial segment to only load into non-special memory. You can do this by setting the "no special memory" bit in the `KIND` field only for the initial segment.

Here's an example of such an initial segment in MPW IIGS format:

```
*********************************************************************
*
* You may wish to do this stuff in the initial segment of your
* application.  The initial segment should be set so that it does not
* load into special memory, or else it is possible that it would load
* into the super hi-res screen.  If this occurred, then QuickDraw II would
* not be able to be started.
*
* Once QuickDraw II is started, the super hi-res screen is taken,
* therefore the rest of the application can not load into it.  Therefore,
* special memory is generally an acceptable place for the rest of the
* application to load, since the special memory needed for the screen
* is already taken.
*
* If the performance of your application would be adversely affected
* by memory fragmentation, then you should also consider purging
* other dormant applications and dormant tools, and then compacting
* memory.  This will prevent fragmentation as much as possible
* while your application is loading.  It also has the cost of longer
* startup time since some tools may have to be reloaded.  This is the
* only way to be sure that tools that you don't want are removed
* from memory before the rest of your application tries to load
* around them.
*
* The Finder is a dormant application when your application is
* launched.  This will cause the Finder to be thrown out of memory,
* and it will have to be reloaded when your application is quit.
*
*********************************************************************

                    case on

                    include 'e16.memory'
                    include 'm16.memory'
                    include 'm16.quickdraw'

screenMode          equ     $80
AppMaxWidth         equ     160             ;Double this and your application
                                            ;will print in BetterText mode.
******************

initialScreen  PROC

myID                equ     1               ;long
zpagehndl           equ     myID+4          ;long

stkAfterLocals equ  zpagehndl+4

directReg           equ     stkAfterLocals
retAddr             equ     directReg+2
passedParms         equ     retAddr+3

                    phd                     ;Set up stack frame.
                    tsc
                    sec
                    sbc     #stkAfterLocals-1
                    tcs
                    tcd
                    pha
                    _MMStartUp
                    pla
                    sta     myID            ;Get the userID
```

```
                    pha
                    _HLockAll           ;Lock down the rest of ourselves, in
                                        ;case we are being restarted.  The
                                        ;loader does not prelock down stuff,
                                        ;so we would be disposing of the rest
                                        ;of ourselves.

                    pea    $1000
                    _PurgeAll           ;Purge other dormant applications.
                                        ;This is optional.
                    pea    $4000
                    _PurgeAll           ;Purge dormant tools.
                                        ;This is optional.

                    _CompactMem         ;Clean up memory.  This is advised.

                    pha                 ;Make direct space for QuickDraw.
                    pha
                    pea    $300>>16     ;Hi-byte of $300 address.
                    pea    $300
                    pei    myID
                    pea    attrLocked+attrFixed+attrPage+attrBank
                    lda    #0
                    pha
                    pha
                    _NewHandle
                    plx
                    stx    zpagehndl
                    plx
                    stx    zpagehndl+2
                    bcc    @a
                    ERRORDEATH 'Out of bank 0 memory'

@a                  lda    zpagehndl
                    sta    >qdstarthndl ;Used for disposing handle at shutdown.
                    txa
                    sta    >qdstarthndl+2
                    lda    [zpagehndl]  ;Start up QuickDraw.  This protects
                    pha                 ;screen RAM from the rest of the
                    pea    screenMode   ;application loading into it.
                    pea    AppMaxWidth
                    pei    myID
                    _QDStartUp
                    bcc    @b
                    ERRORDEATH 'Can''t start up QuickDraw'
@b                                      ;Do title screen here.
                    tsc
                    clc
                    adc    #stkAfterLocals-1
                    tcs
                    pld
                    rtl

qdstarthndl         dc.l   0

                    ENDP
                    END
```

Further Reference:

- *GS/OS Reference*, Volume 1
- *MPW IIGS Tools Reference*
- *APW Assembler Reference*